2011

US DoD Application Domain Empirical Software Cost Analysis

Madachy, Raymond; Boehm, Barry; Clark, Brad; Tan, Thomas; Rosa, Wilson

IEEE

Madachy, Raymond, et al. "Us dod application domain empirical software cost analysis." Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on. IEEE, 2011.

http://hdl.handle.net/10945/60658

US DoD Application Domain Empirical Software Cost Analysis

Raymond Madachy
Department of Systems Engineering
Naval Postgraduate School
Monterey, CA, USA
rjmadach@nps.edu

Barry Boehm, Brad Clark, Thomas Tan
USC Center for Systems and Software Engineering
University of Southern California
Los Angeles, CA, USA
boehm@usc.edu, brad@software-metrics.com,
thomast@usc.edu

Wilson Rosa
Information Technology Division
Air Force Cost Analysis Agency
Arlington, VA, USA
Wilson.Rosa@pentagon.af.mil

***Abstract*— General software cost parameters such as size, effort distribution, and productivity are necessarily imprecise due to variations by domain. To improve this situation, empirical software cost analysis using the primary US DoD cost database has been segmented by domain. This analysis supports a software cost estimation metrics manual for improvements in acquisition policies, procedures and tools. We have addressed the challenges of consistent data definitions and taxonomies across diverse stakeholder communities, data integrity, data formats, and others. We highlight example analysis results from an application domain demonstrating cost estimating relationships, benchmarks on reuse parameters and effort distributions for estimators to use.**

Keywords: software cost estimation, software metrics, software cost models, software productivity, Department of Defense.

I. INTRODUCTION AND BACKGROUND

General software cost parameters such as size, effort distribution, and productivity are necessarily imprecise due to variations by domain. To improve this situation, the Air Force Cost Analysis Agency (AFCAA) in conjunction with service cost agencies, and assisted by the University of Southern California and the Naval Postgraduate School, is conducting domain-oriented empirical research to improve the quality and consistency of estimating methods across cost agencies and program offices through guidance, standardization, and knowledge sharing.

The objectives of this research are to 1) establish a robust and cost effective software metrics collection process and knowledge base that supports the data needs of the United States Department of Defense (US DoD), 2) enhance the utility of the collected data to program oversight and management, and 3) support academic and commercial research into improved cost estimation of future DoD software-intensive systems.

This research will culminate by publishing the *AFCAA Software Cost Estimation Metrics Manual* to help analysts and decision makers develop accurate, easy and quick software cost estimates for avionics, space, ground, and shipboard platforms. The major elements of the manual include:

- Software cost estimation models and their comparisons
- Estimating software size and growth
- Quantifying equivalent size
- Productivity metrics and default cost model inputs
- Quantifying cost uncertainty
- Data collection and normalization.

Our research supports the establishment of policy, related guidance, and recommended implementation approaches for software data collection and analysis across DoD acquisition programs to leverage existing and emerging data standards.

We are also developing an integrated software data repository and related tools for program assessment, cost analysis, software risk assessment, and progress measurement.

A. DoD Empirical Data and Models

The DoD acquisition process outlined in the DoD Instruction 5000.02 policy *Operation of the Defense Acquisition System* mandates the Software Resources Data Report (SRDR) as a regulatory contract reporting requirement [1].

The SRDR is used to obtain both the estimated and actual characteristics of new software developments or upgrades greater than $20M on major contracts and subcontracts. Both the Government program office and later the software contractor submit the SRDR. It constitutes a contract data deliverable for contractors that formalizes the reporting of software metric and resource data.

978-0-7695-4604-9/11 $26.00 © 2011 IEEE
DOI 10.1109/ESEM.2011.56

The SRDR provides a top-level measurement of projects, and our analysis indicates that supplemental data is needed to meet the research goals. We have dealt with a lack of standard reporting for important cost factors related to size, cost attributes, lifecycle and project definition, effort activity reporting, use of common code counting tools consistent with sizing definitions, and quality measures.

Size reporting has many problematic aspects. The units of measurement are not always specified. Size may be measured as logical, physical, etc. Equivalent size parameters for adapted and reused software are also not reported.

Integrating code from previous deliveries (builds, spirals, increments) is not accounted for. Requirements volatility is not reported but it also impacts equivalent size

The equivalent size reported is done in many ways, and one of the goals of this effort was to establish a consistent measurement framework for the size measures.

B. *Data Sources*

The available data sources cover space, ground, air, ship and submarine applications. We currently have over 400 hundred SRDRs. We are normalizing over 300 data points from other sources to include in the analysis including new data from the COCOMO II research [4], the Aerospace Corporation, NRO, Space and Missile Command and others.

We are also collecting specialized datasets for some of the stakeholders, including a space dataset from the NRO. Some of these analyses will not be publicly available due to security constraints.

C. *Data Definitions and Normalization*

Our data definition approach involves 1) review of literature and past research results, 2) updating previous work on cost model comparisons, 3) synthesizing an overall framework that builds on the aforementioned past results, 4) identifying candidate application domains, 5) defining counting rules and standards and 6) validating the framework via trial use with real data.

We have established the following core software size types with standard definitions:

- New
- Adapted
- Reused
- Modified
- Equivalent
- Generated
- Converted
- Off-The-Shelf Software (OTS).

With common definitions the measurements are invariant across cost models and data collections.

A user perspective on Equivalent SLOC (ESLOC) described in [5] is our framework on what to include for size. Equivalent is a way of accounting for relative work done to generate software relative to the code-counted size of the delivered software. We have established rules for what to count for equivalent size in different project environments including technologies (3GLs, 4GLs, automatic code generators) and phases (development vs. maintenance).

Our data normalization strategy involves interviewing program offices and developers to obtain additional information not captured in SRDRs. These include the following items to be consistent with our core data definitions:

- Modification Type (auto generated, re-hosted, translated, modified)
- Source (in-house, third party, prior build, etc.)
- Degree of Modification (% DM, % CM, % IM; SU, UNFM as appropriate)
- Requirements Volatility (% of ESLOC reworked or deleted due to requirements volatility)
- Method (Model Driven Architecture, Object-Oriented, etc.)
- Traditional Cost Model Parameters (True S, SEER, COCOMO, SLIM).

D. *Domain Taxonomy*

Because software productivity tends to vary greatly across applications domains within the DoD, we needed a way to organize analysis results so that we could compare benchmarking results against similar applications. We devised a taxonomy to categorize software productivity results across operating environments and application domains.

Changes to the initial taxonomy resulted from stakeholder workshops and Table 1 shows the current baseline for the project domain taxonomy. The stakeholders have helped to clarify the detailed descriptions of the categories, re-define some and generally broaden our earlier taxonomies.

TABLE I. DOMAIN TAXONOMY

Operating Environments (8)	***Application Domains (24)***	
• Avionics • Business • Manned Ground • Manned Space • Missile and Unmanned Airborne • Shipboard • Unmanned Ground	• Command and Control • Communications • Controls and Displays • Database • Electronics Warfare • Information Assurance • Infrastructure • Internet • Logistics • Maintenance and Diagnostics • Mission Management • Mission • Planning	• Payload • Platform • Process Control • Radars • Scientific Systems • Signal Processing • Simulation and Modeling • Sonar • Test and Evaluation • Tool and Tool Systems • Training • Weapons Delivery and Control

II. DATA ANALYSIS

The section describes elements of our approach to address the empirical challenges and provides interim results from a sample domain.

A. *Cost Estimating Relationship (CER)*

The CER for Communication domain is based on the analysis of 71 observations. The predominate operating environment was Fixed Ground (60) followed by Shipboard (6), Avionics (4) and Unmanned Space (1).

Analysis of the dataset produced the following top level domain CER:

$$PM = A * Size^{B}$$

Where

$A = 2.36 \pm 0.44$

$B = 1.05 \pm 0.11$

Size = Equivalent KSLOC (EKSLOC)

PM = Effort in Person Months (PM)

The upper and lower bounds for A and B represent a 90% confidence interval.

The CER is based on the scatter plot of size versus effort below in Figure 1. Most of the observations are below 100 EKSLOC in size.

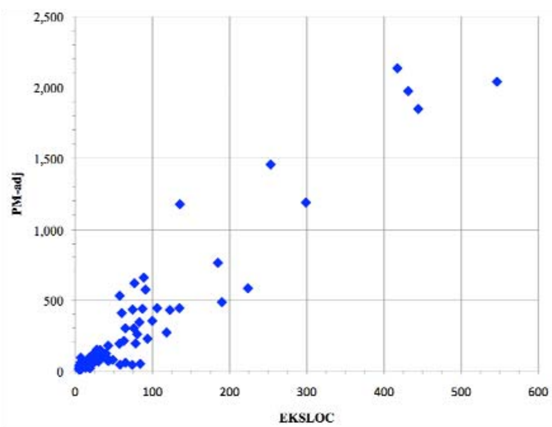

Figure 1. CER Scatter Plot (All Data).

When the 5 to 100 EKSLOC size range is plotted, an anomaly can be observed in the plot. The data below 50 EKSLOC displays a different size-effort relationship than the larger 100+ EKSLOC data in Figure 1, e.g. small components (less than 50 EKSLOC) require large amounts of effort.

The extra effort shown by the smaller size observations is attributable to fixed costs that are independent of the size of the development. Examples include facility and project office startup costs, etc.

The fixed cost is not as observable on larger components because it is amortized over the increasing amount of variable cost, development effort. A *fixed cost function* is used to account for the impact of fixed cost on smaller components.

Observations from 5 to 50 EKSLOC were analyzed using a baseline CER created from observations from 100 to 500 EKSLOC (not the top-level CER). The baseline CER represents variable costs in development. Results from using the baseline CER on the smaller size observations are subtracted from the reported development effort exposing the fixed costs of development. For this domain, the fixed cost function is defined as

$$PM = 42.97 - (0.1543 * EKSLOC).$$

Not all observations below 50 EKSLOC exhibited high costs. The assumption is that small component development that was treated as a stand-alone project incurs additional fixed costs. Conversely, small component development that was part of a larger multi-component system development had the fixed costs amortized over all components.

The analysis of fixed and variable costs across all sizes for this domain shows that fixed costs have the greatest impact for projects below 50 EKSLOC. The data below 50 EKSLOC is adjusted by removing the fixed costs. All the data is then used to create the domain CER shown in the top level equation.

1) *CER Usage and Accuracy*

To estimate effort for a component that has an estimated EKSLOC size, both the domain CER and Fixed Cost function are used. For components smaller than 50 EKSLOC that are part of a multi-component and for components larger than 50 EKSLOC, the top-level domain relationship is used.

The accuracy of the CER is shown using four different measures.

- SEE: The Standard Error of the Estimate is a measure of the difference between the observed and CER estimated effort. The SEE is to linear models as the standard deviation is to a sample mean.
- Bias: The bias is the sum of errors between the observed and estimated effort. The errors, some negative and some positive, should cancel each other.
- R^2: The Coefficient of Determination shows how much variation in effort is explained by size using the domain CER.
- PRED(L): Prediction accuracy is the percentage of CER estimates that are within L percentage of the actual effort observations. L is commonly set to 25%.

TABLE II. CER STATISICS

%SEE	30.6%
Bias	0.0
R^2	0.78
PRED(25)	38%

This analysis also develops benchmarks and estimating guidelines for estimators. These include costing reused and modified software by setting reuse parameters, and how to decompose effort into activity distributions.

B. *Modified SLOC Parameters*

Some observations in the Communications Domain dataset had parameters for converting modified SLOC counts to equivalent SLOC counts. These parameters are Design Modified (DM), Code and Unit Test Modified (CM) and Integration and Test Modified (IM). Figure 2 shows the resulting histogram for DM and Table 3 shows the statistics.

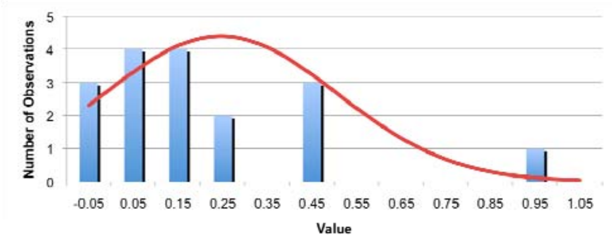

Figure 2. Design Modified (DM) Example Histogram

TABLE III. DESIGN MODIFIED STATISTICS

# Pts	17
Mean	0.25
Median	0.15
Range	0.0 - 1.0
SEE	0.06
90% CL	± 0.11

C. *Effort Distribution*

Effort estimates are decomposed into separate activities for detailed planning. These are expressed as percentage distributions, and used to divide the effort estimate from the domain CER into the different activities. The activities include requirements, architecting and design, code and unit test, integration and test and a catch-all category for other. Figure 3 shows an example distribution for the architecting and design activity. Similar statistics as shown in Table 3 are also reported for the effort distributions.

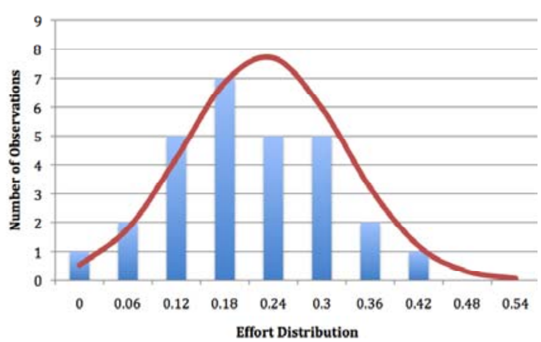

Figure 3. Effort Distribution Example for Architecting and Design

III. CONCLUSIONS AND FUTURE WORK

General software cost parameters such as size, effort distribution, and productivity are necessarily imprecise due to variations by domain. To improve this situation, empirical software cost analysis using the primary US DoD cost database has been segmented by domain.

Our early research identified areas to improve in empirical cost analysis, we re-filtered the dataset after validating the data definition framework and organizational taxonomy, and these are the first example results from rigorously conditioned SRDR data.

We will investigate fixed costs for small projects in the SRDRs and follow-up with data submitters as appropriate, to see if fixed startup costs are included and whether additional form improvements are necessary.

We are also investigating the drivers of effort distributions including size and requirements volatility. For example, small and large projects have different distributions, and added requirements volatility tends to create more proportional effort on back-end test and integration activities.

With more data supplementation to fill in holes and additional projects, we hope to derive additional conclusions from the data for better software cost estimation across the services and commercial industry, and help improve the overall policies, procedures and tools for the Government.

ACKNOWLEDGMENT

This work has been supported by the Air Force Cost Analysis Agency. We also want to acknowledge the help and support from numerous Government and commercial organizations working on this collaborative effort with us.

REFERENCES

[1] United States Department of Defense (DoD), "Instruction 5000.2, Operation of the Defense Acquisition System", December 2008.

[2] W. Rosa, B. Clark, R. Madachy, D. Reifer, and B. Boehm, "Software Cost Metrics Manual", Proceedings of the 42nd Department of Defense Cost Analysis Symposium, February 2009.

[3] B. Boehm, "Future Challenges for Systems and Software Cost Estimation", Proceedings of the 13th Annual Practical Software and Systems Measurement Users' Group Conference, June 2009.

[4] B. Boehm, C. Abts, W. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece, Software Cost Estimation with COCOMO II, Upper Saddle River, NJ: Prentice-Hall, 2000.

[5] R. Stutzke, Estimating Software-Intensive Systems, Upper Saddle River, NJ: Addison Wesley, 2005.

[6] Madachy R, Boehm B, "Comparative Analysis of COCOMO II, SEER-SEM and True-S Software Cost Models", USC-CSSE-2008-816, University of Southern California Center for Systems and Software Engineering, 2008.